

Application Serial No. 09/642,625

**REMARKS**

The Applicant and the undersigned thank Examiner Kiss for his careful review of this application and especially for his time and consideration given during the telephonic interview of November 2, 2004. A summary of this telephonic interview is provided below.

Claims 1-34 have been rejected by the Examiner. Upon entry of this amendment, Claims 1-34 remain pending in this application. The independent claims are Claims 1, 11, 18, 20, 22, 26, and 29.

Consideration of the present application is respectfully requested in light of the above claim amendments to the application, the telephonic interview, and in view of the following remarks.

**Summary of Telephonic Interview of November 2, 2004**

The Applicant and the undersigned thank the Examiner for his time and consideration given during the telephonic interview of November 2, 2004. During this telephonic interview, a proposed amendment to the claims was discussed. The Applicant provided the proposed amendment to the claims in advance of the interview and in connection with an Applicant Initiated Interview Request Form submitted by the Applicant on October 25, 2004.

Examiner Kiss provided his thoughts on the proposed changes to the claims. The Examiner stated that he did not believe that the 1997 article entitled "Understanding Heuristics: Symantec's Bloodhound Technology" (hereinafter, the "UHSBT reference") provides any teaching of generating a behavior pattern for the target program by tracking functions performed and not performed by the target program with flags in a behavior pattern field and by tracking a sequence in which the functions are called by the target program with the behavior pattern field, as was provided in proposed amended independent Claim 11. The undersigned agreed with the Examiner that the UHSBT reference does not provide any discussion of how it tracks behaviors of target programs.

In the proposed amendment submitted on October 25, 2004 to the Examiner, only independent Claims 11, 20, and 26 included the language describing how a behavior pattern is generated with a behavior pattern field. The Examiner and undersigned agreed to amend each of the independent claims to include the language describing the behavior pattern as this feature was not taught by the UHSBT reference. The Examiner stated that while the UHSBT reference

Application Serial No. 09/642,625

does not appear to teach the generation of a behavior pattern as claimed, the Examiner indicated that he would need to conduct an update search before any final decision is made with respect to the claims.

The Applicant and the undersigned request the Examiner to review this interview summary and to approve it by writing "Interview Record OK" along with his initials and the date next to this summary in the margin as discussed in MPEP § 713.04, p. 700-202.

### **Claim Rejections Under 35 U.S.C. § 103**

The Examiner rejected Claims 1, 5, 11, 12, and 19-30 under 35 U.S.C. § 103(a) as being unpatentable over the UHSBT reference in view of a February 2000 article entitled "Enterprise Anti-Virus Software," by Robert Richardson (hereinafter, the "Richardson reference"). The Examiner rejected Claims 2-4 under 35 U.S.C. § 103(a) as being unpatentable over the UHSBT and Richardson references in view of U.S. Pat. No. 6,275,938 issued in the name of Bond et al. (hereinafter, the "Bond reference").

The Examiner rejected Claims 10 under 35 U.S.C. § 103(a) as being unpatentable over the UHSBT reference in view of the Richardson reference, and further in view of a 1995 article entitled "Dynamic Detection and Classification of Computer Viruses Using General Behavior Patterns," authored by Le Charlier et al. (hereinafter, the "Le Charlier reference").

The Examiner rejected Claims 6-9, 13-16, and 31-34 under 35 U.S.C. § 103(a) as being unpatentable over the UHSBT reference in view of the Richardson reference, and further in view of a 1997 article entitled "Blueprint for a Computer Immune System," authored by Jeffrey O. Kephart, et al. (hereinafter, the "Kephart reference"). The Examiner rejected Claim 17 under 35 U.S.C. § 103(a) as being unpatentable over the UHSBT, Richardson, and Kephart references in view of the Le Charlier reference.

The Applicant respectfully offers remarks to traverse these pending rejections. The Applicant will address each independent claim separately as the Applicant believes that each independent claim is separately patentable over the prior art of record.

### **Independent Claim 1**

The rejection of Claim 1 is respectfully traversed. It is respectfully submitted that the UHSBT, Richardson, Bond, Kephart, and the Le Charlier references, individually or in view of

Application Serial No. 09/642,625

each other, fail to describe, teach, or suggest the combination of: (1) initializing a virtual machine within the computer system, the virtual machine comprising a virtual personal computer (PC) implemented by software simulating functionality of a central processing unit and memory and a virtual operating system simulating functionality of a multi-threaded operating system of the computer system; (2) virtually executing a target program within the virtual PC so that the target program interacts only with an instance of the virtual operating system; (3) analyzing behavior of the target program upon completion of virtual execution to identify an occurrence of malicious code behavior based upon an evaluation by the virtual machine of a behavior pattern representing information about all functions simulated by the target program during virtual execution; (4) generating the behavior pattern for the target program by tracking functions performed and not performed by the target program with flags in a behavior pattern field and by tracking a sequence in which the functions are called by the target program with the behavior pattern field; and (5) terminating the virtual PC after the analyzing process, thereby removing from the computer system a copy of the target program that was contained within the virtual PC, as recited in amended independent Claim 1.

#### UHSBT Reference

The UHSBT reference generally describes a heuristic computer virus scanner that can detect computer viruses by using a hybrid system that combines dynamic and static cataloging algorithms. The UHSBT reference describes static heuristic cataloging algorithms as ones that recognize program behaviors by comparing lines of computer code with byte sequences or signatures of a database. See UHSBT reference, page 5, third paragraph. The UHSBT reference describes dynamic heuristic cataloging algorithms as ones that use CPU emulation to gather information regarding a target program. See UHSBT reference, page 6, fourth paragraph. The UHSBT reference also characterizes dynamic heuristic cataloging algorithms as very slow processes that are subject to the whims of the target program being emulated. See UHSBT reference, page 7, fifth paragraph.

The hybrid system of the UHSBT reference uses artificial intelligence technology to isolate and locate the various logical regions of each program it is told to scan. It then analyzes the program logic in each of these components for virus-like behavior. See UHSBT reference, page 10, third paragraph.

Application Serial No. 09/642,625

The UHSBT reference generally discusses the functionality and advantages of its hybrid system without providing sufficient detail to enable one of ordinary skill in the art to construct the technology that is discussed in the article. As evidence that the UHSBT reference is not an enabling reference, the UHSBT reference characterizes its hybrid system as "patent-pending Bloodhound technology." The UHSBT reference further fails to provide any drawings or figures that teach the system components or flow charts that provide steps that describe how the Bloodhound technology is built.

Opposite to the hybrid and dynamic scanners of the UHSBT reference, the invention described by amended independent Claim 1 initializes a virtual machine within a computer system, where the virtual machine comprises a virtual personal computer (PC) implemented by software simulating functionality of a central processing unit and memory and a virtual operating system simulating functionality of a multi-threaded operating system of the computer system. The invention as described by amended independent Claim 1 generates a behavior pattern for the target program by tracking functions performed and not performed by the target program with flags in a behavior pattern field and by tracking a sequence in which the functions are called by the target program with the behavior pattern field. The UHSBT reference does not provide any teaching of such a behavior pattern as recited in amended independent Claim 1 in combination with the other claim elements.

#### The Richardson Reference

The Examiner admits that the UHSBT reference fails to provide any teaching of a virtual operating system simulating functionality of a multi-threaded operating system. To make up for this deficiency, the Examiner relies upon the Richardson reference.

Similar to the UHSBT reference, the Richardson reference is an article that discusses enterprise anti-virus software in general without describing any particular software in detail that would enable one of ordinary skill in the art to build any of the systems that are mentioned in the reference. The Richardson reference generally mentions the advantages and disadvantages of enterprise anti-virus software.

The Examiner relies on the nineteenth paragraph of the Richardson reference to provide a teaching of a multi-threaded operating system. This paragraph of the Richardson reference does not provide any detail of how to construct or build a virtual operating system simulating

Application Serial No. 09/642,625

functionality of a multi-threaded operating system. The nineteenth paragraph of the Richardson reference is reproduced as follows:

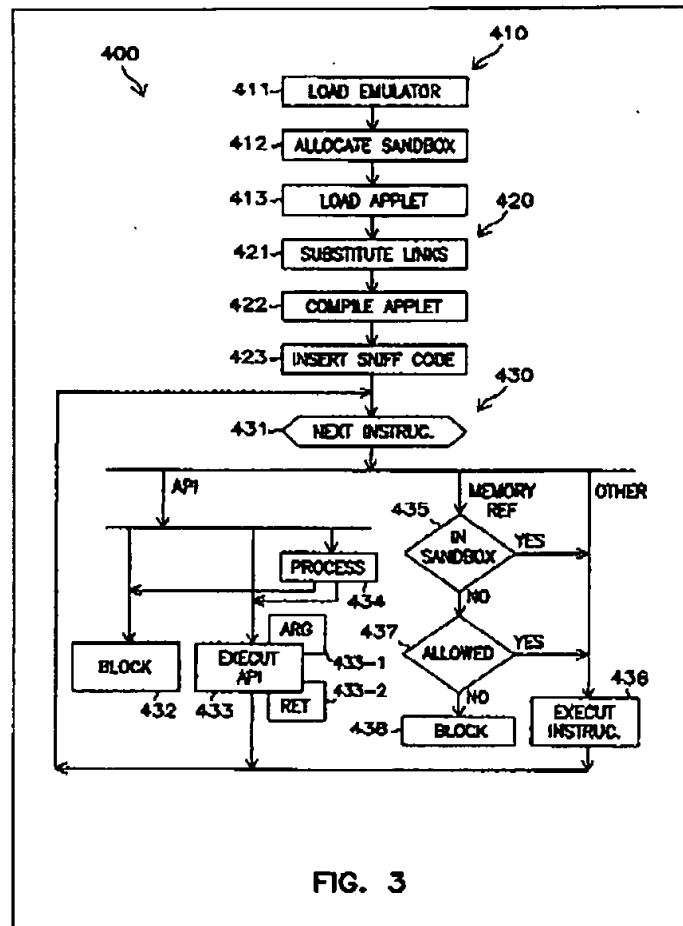
The second method for behavior acquisition is emulation. That's a bit trickier, according to Nachenberg. 'You run the program through a Windows emulator or a Macintosh or a Word macroemulator, and you watch what the program does, and then you catalog all the behavior. However, there are some issues with that technique because it's basically subject to the whims of the virus. For instance, if the virus only wants to format your hard drive on January 15 at 9 p.m., and you're emulating that virus in a simulator and have set the time for January 14, the virus is never going to display that format behavior.'

Even if the Examiner's proposed combination of the UHSBT and Richardson references was feasible and enabling to one of ordinary skill in the art, the proposed combination does not provide any teaching of generating a behavior pattern for the target program by tracking functions performed and not performed by the target program with flags in a behavior pattern field and by tracking a sequence in which the functions are called by the target program with the behavior pattern field, as recited in amended independent Claim 1.

#### The Bond Reference

The Examiner admits that the UHSBT reference fails to provide any teaching of a virtual operating system simulating functionality of an operating system application program interface, wherein virtual execution of the target program causes the target program to interact with the simulated operating system application programming interface. To make up for this deficiency of the UHSBT reference, the Examiner relies upon the Bond reference.

The Bond reference generally describes a sandbox or pre-allocated memory range in which untrusted computer code is loaded, run, and observed. Figure 3 of the Bond reference illustrates major steps 400 of one embodiment of the sandbox technology that allows applets executed on a personal computer to access all of an operating systems services without compromising a PC's security. See Figure 3 of the Bond reference reproduced below.



Step 421 of the method illustrated above in Figure 3 of the Bond reference substitutes an applet's static links with links to thunk modules. Thunks are translation-code modules that allow executable code to access the host operating system, while preventing breaches of the host system's security. A thunk can determine whether an API call is one which should be allowed to execute on the operating system or not. See the Bond reference, column 2, lines 5-14 and in column 6, lines 24-50.

Similar to the UHSBT reference, the Bond reference also fails to provide any teaching of generating a behavior pattern for a target program by tracking functions performed and not performed by the target program with flags in a behavior pattern field and by tracking a sequence in which the functions are called by the target program with the behavior pattern field, as recited in amended independent Claim 1.

Application Serial No. 09/642,625

The Le Charlier Reference

The Examiner admits that the UHSBT reference fails to provide any teaching of tracking an order in which functions are virtually executed by a target program within a virtual PC to provide a complete record of all functions simulated by the target program, as if the target program were executed on the computer system. To make up for this deficiency of the UHSBT reference, the Examiner relies upon the Le Charlier reference.

The Le Charlier reference describes how an emulator is used to monitor system activity of a virtual PC, and how an expert system ASAX is used to analyze the stream of data that the emulator produced. The Le Charlier reference also describes how general rules are used to generally detect real viruses reliably, and specific rules to extract details of their behavior. See the Le Charlier reference, page 1.

Sections 4.5 and 4.6 of the Le Charlier reference describe an audit trail of an emulator. The audit trail includes parameters popped from a stack with return values. See Figure 3 of the Le Charlier reference reproduced below and page 15.

```
<CS=3911 Type=0 Pn=30 arg() ret( AX=5)>
<CS=3911 Type=0 Pn=29 arg() ret( BX=128 ES=3911)>
<CS=3911 Type=0 Pn=64 arg( AL=61 CL=3 str1=.COM) ret( AL=0 CP=0)>
<CS=3911 Type=0 Pn=51 arg( AL=0 str1=CONRAD.COM) ret( AL=0 CX=32 CP=0)>
<CS=3911 Type=0 Pn=51 arg( AL=1 str1=CONRAD.COM) ret( AL=0 CX=32 CP=0)>
<CS=3911 Type=0 Pn=45 arg( AL=2 CL=32 str1=COMMAND.COM) ret( AL=0 AX=5 CP=0)>
<CS=3911 Type=0 Pn=73 arg( BX=5) ret( CX=10241 DX=6208 CP=0)>
<CS=3911 Type=0 Pn=27 arg() ret( CX=5121 DX=8032)>
<CS=3911 Type=0 Pn=67 arg( BX=5 CX=3 DX=828 DS=3911) ret( AX=3 CP=0)>
<CS=3911 Type=0 Pn=50 arg( AL=2 BX=5 CX=0 DX=0) ret( AL=0 AX=50031 DX=0 CP=0)>
<CS=3911 Type=0 Pn=48 arg( BX=5 CX=848 DX=313 DS=3911) ret( AX=848 CP=0)>
<CS=3911 Type=0 Pn=50 arg( AL=0 BX=5 CX=0 DX=0) ret( AL=0 AX=0 DX=0 CP=0)>
<CS=3911 Type=0 Pn=48 arg( BX=5 CX=3 DX=831 DS=3911) ret( AX=3 CP=0)>
<CS=3911 Type=0 Pn=74 arg( BX=5 CX=10271 DX=6206) ret( CP=0)>
<CS=3911 Type=0 Pn=46 arg( BX=6) ret( CP=0)>
<CS=3911 Type=0 Pn=51 arg( AL=1 str1=CONRAD.COM) ret( AL=0 CX=32 CP=0)>
:
```

Figure 3: Excerpt from an audit trail for the Vienna virus

Figure 3 of the Le Charlier reference is an example of an audit trail. The audit trail of this figure is a human representation of a binary NADF file. The example illustrated is from the Vienna virus.

Application Serial No. 09/642,625

Similar to the UHSBT reference, the Le Charlier reference also fails to provide any teaching of generating a behavior pattern for a target program by tracking functions performed and not performed by the target program with flags in a behavior pattern field and by tracking a sequence in which the functions are called by the target program with the behavior pattern field, as recited in amended independent Claim 1.

#### The Kephart Reference

The Examiner admits that the UHSBT and Richardson references fail to provide any teaching of determining that a target program has been modified; analyzing the modified program to provide a second behavior pattern; comparing the behavior pattern to the second behavior pattern to determine if malicious code is present in the modified program; generating a new behavior pattern each time the target program is modified; identifying altered bits indicating an addition of an infection procedure; and identifying the behavior pattern as a match to the second behavior pattern when the modified target program is a new version of the target program. To make up for these numerous deficiencies of the UHSBT and Richardson references, the Examiner relies upon the Kephart reference.

The Kephart reference describes an immune system for computers that sense the presence of a previously unknown pathogen, and within minutes, automatically derives and deploys a prescription for detecting and removing it. The system is integrated with a commercial anti-virus product. See the Kephart reference, abstract.

Section 3.1.1 of the Kephart reference describes a generic disinfection technique and a classifier that recognizes potential viruses based on characteristic machine-code features. With the generic disinfection technique, when software is first installed, a fingerprint of each executable file is computed and stored in a database. The fingerprint consists of less than 100 bytes of information about each program F, including its size, date, and various checksums. On subsequent scans of the system, the fingerprint is recomputed and compared with the stored one. If there has been a change, the system is scanned for known viruses. See the Kephart reference, page 5, third paragraph.

Similar to the UHSBT and Richardson references, the Kephart reference also fails to provide any teaching of generating a behavior pattern for a target program by tracking functions performed and not performed by the target program with flags in a behavior pattern field and by



Application Serial No. 09/642,625

tracking a sequence in which the functions are called by the target program with the behavior pattern field, as recited in amended independent Claim 1.

#### Conclusion Regarding Independent Claim 1

In light of the differences between Claim 1 and the UHSBT, Richardson, Bond, Kephart, and the Le Charlier references mentioned above, one of ordinary skill in the art recognizes that the combination proposed by the Examiner cannot anticipate or render obvious the recitations as set forth in amended independent Claim 1. Accordingly, reconsideration and withdrawal of this rejection of Claim 1 are respectfully requested.

#### Independent Claim 11

The rejection of Claim 11 is respectfully traversed. It is respectfully submitted that the UHSBT, Bond, Richardson, Kephart, and the Le Charlier references, individually or in view of each other, fail to describe, teach, or suggest the combination of: (1) initializing a virtual machine within the computer system, the virtual machine comprising software simulating functionality of a central processing unit and memory and a virtual operating system simulating functionality of a multi-threaded operating system of the computer system; (2) virtually executing a target program with the virtual machine so that the target program interacts with an instance of the virtual operating system rather than with the operating system of the computer system, whereby the malicious code is fully executed during virtual execution of the target program if the target program comprises the malicious code; (3) generating a behavior pattern for the target program by tracking functions performed and not performed by the target program with flags in a behavior pattern field and by tracking a sequence in which the functions are called by the target program with the behavior pattern field in order to collect information about all functions simulated by the target program during virtual execution; and (4) terminating the virtual machine upon completion of the virtual execution of the target program, leaving behind a record of the behavior pattern that is representative of operations of the target program with the computer system, including operations of the malicious code if the target program comprises the malicious code, as recited in amended independent Claim 11.

Similar to the analysis of independent Claim 1, the Examiner's proposed combination of references fails to address the specifics of initializing a virtual machine within the computer

Application Serial No. 09/642,625

system, the virtual machine comprising software simulating functionality of a central processing unit and memory and a virtual operating system simulating functionality of a multi-threaded operating system of the computer system. The UHSBT, Bond, Richardson, Kephart, and the Le Charlier references fail to provide any teaching of generating a behavior pattern for the target program by tracking functions performed and not performed by the target program with flags in a behavior pattern field and by tracking a sequence in which the functions are called by the target program with the behavior pattern field in order to collect information about all functions simulated by the target program during virtual execution, as recited in amended independent Claim 11.

In light of the differences between Claim 11 and the UHSBT, Richardson, Bond, Kephart, and the Le Charlier references mentioned above, one of ordinary skill in the art recognizes that the combination proposed by the Examiner cannot anticipate or render obvious the recitations as set forth in amended independent Claim 11. Accordingly, reconsideration and withdrawal of this rejection of Claim 11 are respectfully requested.

#### Independent Claim 18

The rejection of Claim 18 is respectfully traversed. It is respectfully submitted that the UHSBT, Bond, Richardson, Kephart, and the Le Charlier references, individually or in view of each other, fail to describe, teach, or suggest the combination of: (1) initializing a virtual machine for the computer system, the virtual machine comprising a virtual personal computer (PC) implemented by software simulating functionality of a central processing unit memory a virtual operating system simulating functionality of a multi-threaded operating system of the computer system; (2) executing a target program within the virtual PC so that the target program completes a virtual execution by interacting only with an instance of the virtual operating system; (3) generating a behavior pattern by completing virtual execution of the target program within the virtual PC and by tracking functions performed and not performed by the target program with flags in a behavior pattern field and by tracking a sequence in which the functions are called by the target program, the behavior pattern representative of operational functions completed by the target program during virtual execution, including at least one of virtual operating system calls, Input/Output functions and program functions supported by the target program; (4) upon completion of virtual execution, operating the virtual machine to compare the

Application Serial No. 09/642,625

behavior pattern generated by virtual execution of the target program to a behavior pattern representative of operations by the malicious code to identify an occurrence of malicious code behavior; and (5) in the event that the comparison process results in a match representing an identification of malicious code behavior by the target program, then identifying the target program as comprising the malicious code, as recited in amended independent Claim 18.

Similar to the analysis of independent Claim 1, the Examiner's proposed combination of references fails to address the specifics of (1) initializing a virtual machine for the computer system, the virtual machine comprising a virtual personal computer (PC) implemented by software simulating functionality of a central processing unit memory a virtual operating system simulating functionality of a multi-threaded operating system of the computer system. The UHSBT, Bond, Richardson, Kephart, and the Le Charlier references fail to provide any teaching of generating a behavior pattern by completing virtual execution of the target program within the virtual PC and by tracking functions performed and not performed by the target program with flags in a behavior pattern field and by tracking a sequence in which the functions are called by the target program, the behavior pattern representative of operational functions completed by the target program during virtual execution, including at least one of virtual operating system calls, Input/Output functions and program functions supported by the target program, as recited in amended independent Claim 18.

In light of the differences between Claim 18 and the UHSBT, Richardson, Bond, Kephart, and the Le Charlier references mentioned above, one of ordinary skill in the art recognizes that the combination proposed by the Examiner cannot anticipate or render obvious the recitations as set forth in amended independent Claim 18. Accordingly, reconsideration and withdrawal of this rejection of Claim 18 are respectfully requested.

#### Independent Claim 20

The rejection of Claim 20 is respectfully traversed. It is respectfully submitted that the UHSBT, Bond, Richardson, Kephart, and the Le Charlier references, individually or in view of each other, fail to describe, teach, or suggest the combination of: (1) executing a target program within a virtual personal computer (PC) so that the target program completes a virtual execution by interacting only with an instance of a virtual operating system, the virtual PC comprising software operative to simulate functionality of a processor and memory, the virtual

Application Serial No. 09/642,625

operating system operative to simulate functionality of a multi-threaded operating system for the computer system, the virtual PC and the virtual operating system operating in combination to form a virtual machine; (2) collecting information about the behavior of the target program during virtual execution of the target program by the virtual machine by tracking functions performed and not performed by the target program with flags in a behavior pattern field and by tracking a sequence in which the functions are called by the target program in order to create a record of virtual operations of the target program, whereby the record reflects a plurality of operations of the malicious code if the target program comprises the malicious code; (3) upon completion of virtual execution of the target program, analyzing the record with the virtual machine to identify an occurrence of malicious code behavior by comparing the record to a behavior pattern representative of the operations performed by the malicious code; and (4) in the event that the record matches the malicious code behavior, then identifying the target program as comprising the malicious code., as recited in amended independent Claim 20.

Similar to the analysis of independent Claim 1, the Examiner's proposed combination of references fails to address the specifics of executing a target program within a virtual personal computer (PC) so that the target program completes a virtual execution by interacting only with an instance of a virtual operating system, the virtual PC comprising software operative to simulate functionality of a processor and memory, the virtual operating system operative to simulate functionality of a multi-threaded operating system for the computer system, the virtual PC and the virtual operating system operating in combination to form a virtual machine. The UHSBT, Bond, Richardson, Kephart, and the Le Charlier references fail to provide any teaching of collecting information about the behavior of the target program during virtual execution of the target program by the virtual machine by tracking functions performed and not performed by the target program with flags in a behavior pattern field and by tracking a sequence in which the functions are called by the target program in order to create a record of virtual operations of the target program, whereby the record reflects a plurality of operations of the malicious code if the target program comprises the malicious code, as recited in amended independent Claim 20.

In light of the differences between Claim 20 and the UHSBT, Richardson, Bond, Kephart, and the Le Charlier references mentioned above, one of ordinary skill in the art recognizes that the combination proposed by the Examiner cannot anticipate or render obvious

Application Serial No. 09/642,625

the recitations as set forth in amended independent Claim 20. Accordingly, reconsideration and withdrawal of this rejection of Claim 20 are respectfully requested.

Independent Claim 22

The rejection of Claim 22 is respectfully traversed. It is respectfully submitted that the UHSBT, Bond, Richardson, Kephart, and the Le Charlier references, individually or in view of each other, fail to describe, teach, or suggest the combination of: (1) virtually executing a target program within a virtual machine comprising a virtual personal computer (PC) implemented by software operative to simulate functionality of a processor, and memory and a virtual operating system having software simulating functionality of a multi-threaded operating system for the computer system wherein virtual execution of the target program comprises interactions with an instance of the virtual operating system; (2) creating a record of all functions simulated by the target program during virtual execution of the target program by the virtual machine, the record comprising a behavior pattern representative of the behavior of the target program as if it were executed on the computer system, the behavior pattern comprising characteristics of malicious code behavior in the event that the target program comprises the malicious code; and (3) creating the behavior pattern by tracking functions performed and not performed by the target program with flags in a behavior pattern field and by tracking a sequence in which the functions are called by the target program with the behavior pattern field, as recited in amended independent Claim 22.

Similar to the analysis of independent Claim 1, the Examiner's proposed combination of references fails to address the specifics of (1) virtually executing a target program within a virtual machine comprising a virtual personal computer (PC) implemented by software operative to simulate functionality of a processor, and memory and a virtual operating system having software simulating functionality of a multi-threaded operating system for the computer system wherein virtual execution of the target program comprises interactions with an instance of the virtual operating system. The UHSBT, Richardson, Bond, Kephart, and the Le Charlier references fail to provide any teaching of creating the behavior pattern by tracking functions performed and not performed by the target program with flags in a behavior pattern field and by tracking a sequence in which the functions are called by the target program with the behavior pattern field, as recited in amended independent Claim 22.

Application Serial No. 09/642,625

In light of the differences between Claim 22 and the UHSBT, Richardson, Bond, Kephart, and the Le Charlier references mentioned above, one of ordinary skill in the art recognizes that the combination proposed by the Examiner cannot anticipate or render obvious the recitations as set forth in amended independent Claim 22. Accordingly, reconsideration and withdrawal of this rejection of Claim 22 are respectfully requested.

Independent Claim 26

The rejection of Claim 26 is respectfully traversed. It is respectfully submitted that the UHSBT, Bond, Richardson, Kephart, and the Le Charlier references, individually or in view of each other, fail to describe, teach, or suggest the combination of: (1) executing a target program within a virtual personal computer (PC) so that the target program completes a virtual execution by interacting only with an instance of a virtual operating system, the virtual PC comprising software operative to simulate functionality of a processor and memory, the virtual operating system operative to simulate functionality of a multi-threaded operating system for the computer system, the virtual PC and the virtual operating system operating in combination to form a virtual machine; (2) collecting information about the behavior of the target program in response to virtual execution of the target program by the virtual machine; (3) in response to completing virtual execution of the target program, collecting information about interrupt call operations that call any interrupt service routine modified by the virtual execution of the target program; (4) creating a record by tracking functions performed and not performed by the target program with flags in a behavior pattern field and by tracking a sequence in which the functions are called by the target program with the behavior pattern field, the functions comprising the interrupt call operations, the record comprising the information collected about the virtual execution of the target program and the interrupt call operations that call any interrupt service routine modified by the virtual execution of the target program; (5) analyzing the record to identify an occurrence of malicious code behavior by comparing the record to a behavior pattern representative of the operations performed by the malicious code; and (6) in the event that the record matches the malicious code behavior, then identifying the target program as comprising the malicious code, as recited in amended independent Claim 26.

Similar to the analysis of independent Claim 1, the Examiner's proposed combination of references fails to address the specifics of executing a target program within a virtual personal

Application Serial No. 09/642,625

computer (PC) so that the target program completes a virtual execution by interacting only with an instance of a virtual operating system, the virtual PC comprising software operative to simulate functionality of a processor and memory, the virtual operating system operative to simulate functionality of a multi-threaded operating system for the computer system, the virtual PC and the virtual operating system operating in combination to form a virtual machine. The UHSBT, Richardson, Bond, Kephart, and the Le Charlier references fail to provide any teaching of creating a record by tracking functions performed and not performed by the target program with flags in a behavior pattern field and by tracking a sequence in which the functions are called by the target program with the behavior pattern field, the functions comprising the interrupt call operations, the record comprising the information collected about the virtual execution of the target program and the interrupt call operations that call any interrupt service routine modified by the virtual execution of the target program, as recited in amended independent Claim 26.

In light of the differences between Claim 26 and the UHSBT, Richardson, Bond, Kephart, and the Le Charlier references mentioned above, one of ordinary skill in the art recognizes that the combination proposed by the Examiner cannot anticipate or render obvious the recitations as set forth in amended independent Claim 26. Accordingly, reconsideration and withdrawal of this rejection of Claim 26 are respectfully requested.

#### Independent Claim 29

The rejection of Claim 29 is respectfully traversed. It is respectfully submitted that the UHSBT, Bond, Richardson, Kephart, and the Le Charlier references, individually or in view of each other, fail to describe, teach, or suggest the combination of: (1) initializing a virtual machine for the computer system, the virtual machine comprising a virtual personal computer (PC) implemented by software simulating functionality of a central processing unit and memory and a virtual operating system simulating functionality of a multi-threaded operating system of the computer system, the initializing step comprising the steps of extracting the file structure of a target program and loading the target program into the software-simulated memory of the virtual PC; (2) executing a target program within the virtual PC so that the target program completes a virtual execution by interacting only with an instance of the virtual operating system; (3) generating a behavior pattern by completing virtual execution of the entire code of the target program within the virtual PC and by tracking functions performed and not performed by the

Application Serial No. 09/642,625

target program with flags in a behavior pattern field and by tracking a sequence in which the functions are called by the target program, the behavior pattern representative of a sequence of operational functions completed by the target program during virtual execution, including at least one of virtual operating system calls, Input/Output functions and program functions supported by the target program; (4) upon completion of virtual execution, operating the virtual machine to compare the behavior pattern generated by virtual execution of the target program to a behavior pattern representative of operations by the malicious code to identify an occurrence of malicious code behavior; and (5) in the event that the comparison process results in a match representing an identification of malicious code behavior by the target program, then identifying the target program as comprising the malicious code, as recited in amended independent Claim 29.

Similar to the analysis of independent Claim 1, the Examiner's proposed combination of references fails to address the specifics of (1) initializing a virtual machine for the computer system, the virtual machine comprising a virtual personal computer (PC) implemented by software simulating functionality of a central processing unit and memory and a virtual operating system simulating functionality of a multi-threaded operating system of the computer system, the initializing step comprising the steps of extracting the file structure of a target program and loading the target program into the software-simulated memory of the virtual PC. The UHSBT, Richardson, Bond, Kephart, and the Le Charlier references fail to provide any teaching of generating a behavior pattern by completing virtual execution of the entire code of the target program within the virtual PC and by tracking functions performed and not performed by the target program with flags in a behavior pattern field and by tracking a sequence in which the functions are called by the target program, the behavior pattern representative of a sequence of operational functions completed by the target program during virtual execution, including at least one of virtual operating system calls, Input/Output functions and program functions supported by the target program, as recited in amended independent Claim 29.

In light of the differences between Claim 29 and the UHSBT, Richardson, Bond, Kephart, and the Le Charlier references mentioned above, one of ordinary skill in the art recognizes that the combination proposed by the Examiner cannot anticipate or render obvious the recitations as set forth in amended independent Claim 29. Accordingly, reconsideration and withdrawal of this rejection of Claim 29 are respectfully requested.



Application Serial No. 09/642,625

Dependent Claims 2-10, 12-17, 19, 21, 23-25, 27-28, and 30-34

The Applicant respectfully submits that the above-identified dependent claims are allowable because the independent claims from which they depend are patentable over the cited references. The Applicant also respectfully submits that the recitations of these dependent claims are of patentable significance.


In view of the foregoing, the Applicant respectfully requests that the Examiner withdraw the pending rejections of dependent Claims 2-10, 12-17, 19, 21, 23-25, 27-28, and 30-34.

CONCLUSION

The foregoing is submitted as a full and complete response to the Office Action mailed on June 29, 2004. The Applicant and the undersigned thank Examiner Kiss for consideration of these remarks. The Applicant has amended the claims and has submitted remarks to traverse rejections of Claims 1-34. The Applicant respectfully submits that the present application is in condition for allowance. Such action is hereby courteously solicited.

If the Examiner believes that there are any issues that can be resolved by a telephone conference, or that there are any formalities that can be corrected by an Examiner's amendment, please contact the undersigned in the Atlanta Metropolitan area (404) 572-2884.

Respectfully submitted,

  
Steven P. Wigmore  
Reg. No. 40,447

King & Spalding LLP  
191 Peachtree Street, N.E.  
Atlanta, Georgia 30303-1763  
telephone: (404) 572.4600  
K&S File No. 05456-105041